Soaplab
SOAP-based Analysis Web Services

Martin Senger <senger@ebi.ac.uk>
http://industry.ebi.ac.uk/soaplab

What is the task?
1. You have (or someone else has) one or more command-line analysis tools
2. You want to access them from other computers, eventually over an Internet
3. You want to access them using your (or someone else’s) programs, not just by filling the forms and clicking on web pages

How it fits with the rest of the world
• Phase 1: Describe your command-line tool (preferable using an ACD file)
  – http://www.hgmp.mrc.ac.uk/Software/EMBOSS/Acd/index.html
• Phase 2: Soaplab
  – this whole presentation is about this phase
• Phase 3: Write your own programs (GUIs and others) to access tools via Soaplab
  – part of this presentation is about this
  – Talisman: an example of a web GUI presenting and using Soaplab (and many more things)
    • http://www.ebi.ac.uk/talisman/index.html

Graphically...

ACD – An EMBOSS Application

ACD – Non-EMBOSS Application

Important:
• how to write ACD
• what API to use by the clients

Download:
• applab-cembalo-soaplab-version
• analysis-interfaces-version
• soaplab-clients-version
http://industry.ebi.ac.uk/soaplab/dist/
What Web Services?

- Analysis Factory Service (misnomer)
  - produces a list of all available services
- Analysis Service
  - represents one analysis tool and allows to start it, control it, and to exploit it
  - does it in loosely typed manner using the same API for all analyses
- Derived Analysis Service
  - does the same as the Analysis Service above but in a strongly typed manner having individual API for any individual analysis tool

Soaplab API
http://industry.ebi.ac.uk/soaplab/API.html

The main methods for running an analysis

- createJob (inputs)
- createAndRun (inputs)
- waitFor ($job_id);
- result ($job_id)
- getElapsed ($job_id)
- getStarted ($job_id)
- getCreated ($job_id)
- getLastEvent ($job_id)
- getStatus ($job_id)

The methods for asking about the whole analysis

- describe (service)
- discover (service)
- getAvailableCategories (service)
- getAvailableAnalysesInCategory (service)
- getServiceLocation (service)
- set_firstonly (datum)
- set_format (datum)
- set_sequence_usa (datum)

An example of (additional) methods for a derived service

- result ($job_id)
- getCharacteristics ($job_id)
- getAvailableCategories ($job_id)
- getResultSpec ($job_id)
- getAnalysisType ($job_id)
- getSomeResults ($job_id)
- destroy ($job_id)
- getResults ($job_id)

What names to use for inputs?

- Ask (getInputSpec) – expected types: basic types
- And the results? Ask (getResultSpec) – expected types: basic types, byte[ ], byte[ ]

A. Client developer

- Needs to know the Soaplab API
- Can develop in any programming language
- Needs to know a place (an endpoint) with a running Soaplab services
  - http://industry.ebi.ac.uk/soap/soaplab (EBSiSS)
- Can use the ready, pre-prepared Java and Perl clients

Once you know the API...

- Direct usage of the Soaplab API...
  - supported types: basic types
- ...or using BioPerl modules
  - expected types: basic types, byte[ ], byte[ ]
  - better check org.embl.ebi.SoaplabClient.Api1/Api2/Api3
What else…

• Code examples at:
  – http://industry.ebi.ac.uk/soaplab/UserGuide.html
• Play also with existing clients:
  java AnalysisClient –h
  java FactoryClient –h
  java RegistryAdmin –h
  java TutorialTest

B. Service provider

• EMBOSS provider
  – contains pre-generated XML files for all EMBOSS analysis (but not EMBOSS itself)
• Provider of your own analysis tools
  – contains a2dxml for creating XML files

Installation

• Run perl INSTALL.pl
  – and you will be asked many questions
  – or prepare your answers in environment variables and run: perl INSTALL.pl –q
• To add your own analyses?
  – write an ACD file (e.g. metadata/bigdeal.acd)
  – run: generator/acd2xml –d –l MyApps.xml bigdeal
  – see generator/README for details

What questions to expect?

<table>
<thead>
<tr>
<th>basics</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAVA_HOME</td>
</tr>
<tr>
<td>PROJECT_HOME</td>
</tr>
<tr>
<td>RESULTS_URL</td>
</tr>
<tr>
<td>tomcat and axis locations</td>
</tr>
<tr>
<td>TOMCAT_HOME</td>
</tr>
<tr>
<td>TOMCAT_URL</td>
</tr>
<tr>
<td>AXIS_LIB</td>
</tr>
<tr>
<td>AXIS_IN_TOMCAT</td>
</tr>
<tr>
<td>mySQL locations</td>
</tr>
<tr>
<td>MYSQL_URL</td>
</tr>
<tr>
<td>MYSQL_USER</td>
</tr>
<tr>
<td>MYSQL_PASSWD</td>
</tr>
<tr>
<td>ALWEB_SCRIPTS_URL</td>
</tr>
<tr>
<td>ALWEB_DOCS_URL</td>
</tr>
<tr>
<td>ADD_TO_PATH</td>
</tr>
<tr>
<td>nothing to do with web-services (so ignore it…)</td>
</tr>
</tbody>
</table>

And run it!

• ./run-AppLab-server
  – check AppLabServer.log
• (start your Tomcat)
• ./ws/deploy-web-services
• try any client
  – check soaplab.log
• …complain to senger@ebi.ac.uk

Deploying services

• That can be tricky…because it includes many things:
  – move your classes to Tomcat
  – create service descriptors (files *.wsdd)
  – generate and compile code for the derived services
  – etc.
• Therefore, use a Soaplab admin client:
  – java AAdmin –help
  – which is used by ws/deploy-web-services, anyway